**RJ10486 (A1108-005) August 3, 2011**
**Computer Science**

# IBM Research Report

## Towards Conversational Interfaces to Web Applications

**Tessa Lau[1], Julian Cerruti[2], Morgan Dixon[3], Jeffrey Nichols[1]**

[1]IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA  95120-6099
USA

[2]IBM Argentina
Ing. Butty 275 - C1001AFA
Buenos Aires, Argentina

[3]Computer Science and Engineering
DUB Group
University of Washington

**Research Division**
**Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich**

# Towards conversational interfaces to web applications

Tessa Lau[†], Julian Cerruti[⋆], Morgan Dixon[×], Jeffrey Nichols[†]

[†]IBM Research – Almaden
650 Harry Road
San Jose, CA 95120 USA

[⋆]IBM Argentina
Ing. Butty 275 - C1001AFA
Buenos Aires, Argentina

[×]Computer Science &
Engineering
DUB Group, University of
Washington

tessalau@us.ibm.com

## ABSTRACT

Today's conversational interfaces are largely based on the paradigm of information retrieval from databases. In this position paper, we propose a radically different approach: building CIs on top of existing web applications. Such a system will draw together research in task modeling, web usage mining, information extraction, as well as the vast amount of existing research on traditional CIs.

## 1. INTRODUCTION

Conversational interfaces are an alternative to traditional graphical interfaces as a means for people to interact with computers. As computing moves to smaller mobile devices with limited I/O capabilities, conversational interfaces increase users' ability to access information while on the move – anytime, anywhere. Moreover, conversational interfaces could also enable persons with disabilities access to IT services that they were formerly unable to access.

Most past work on conversational interfaces has focused on the information retrieval task of selecting and filtering records from a database [21]. While focusing on database queries makes the problem more tractable, it also limits the applicability of conversational interfaces. Despite decades of research, conversational interfaces have only achieved success in limited domains; no general-purpose conversational interface has yet emerged.

In this position paper, we propose a radically different approach to creating conversational interfaces: building them *on top of* existing web applications. Can we automatically create general-purpose conversational interfaces that enable users to do anything they could do on the web, through conversation?

At first glance, creating conversational interfaces for arbitrary web applications is tremendously challenging. Every website is different; there is a enormous variation in the types of tasks that can be accomplished using the web. The web is designed for people to use visually; the state of the art in screenreaders for blind users [5] is still very cumbersome. Crafting easy-to-use conversational interfaces seems to require custom programming specific to each website.

Yet the amount of metadata inherent in web applications leads us to believe that automatic construction of conversational interfaces (CIs) for arbitrary websites may be possible.

Open web standards such as HTML and CSS enable systems to programmatically "see inside" the structure of web applications. The structure and organization of the application (e.g. the page layout, the link hierarchy) give clues as to what can be done in the application. This organization, created by the application designer, guides users along paths that enable them to accomplish specific tasks. We believe that this open model, combined with recent advances in web usage mining, information extraction, and task modeling, enable the creation of rich, interactive CIs based on existing web applications.

The remainder of this paper is organized as follows. First we provide a brief overview of prior work in database-oriented conversational interfaces. Then we explore in more depth the differences between database-backed CIs and web-based CIs. Finally, we conclude with observations and directions for future work.

## 2. PRIOR WORK

Conversational interfaces have been around for a long time; Zue and Glass [21] provide a good survey of the literature. Due to the difficulty of developing general-purpose CIs, they tend to be targeted at a specific domain, such as train schedules [1], restaurants [18], job listings [19], or call routing [7].

The predominant paradigm for CIs assumes that they are interfaces to a database and that the user is performing an information retrieval task from the database. Several aspects of database queries make the problem of building conversational interfaces to databases relatively tractable: the database schema is typically known; possible data values can be computed; and the query language (e.g., SQL) is standardized with known semantics.

Early research in CIs investigated how to make them easily *transportable* across arbitrary databases with little programming effort [8]. Systems such as SpeechBuilder [6] proposed a framework to make CIs easier to develop by plugging together existing components. Despite such efforts, CIs are still not in wide use for general-purpose tasks such as those found on most websites.

The database interface also enables developers to provide theoretical guarantees about the completeness of a CI. For example, Popescu *et al* [19] have proven that a particular class of queries can be mapped precisely to SQL queries.

On the web, so such guarantees are available.

More recently, conversational interfaces such as Siri[1] and Google's Talk Guru[2] provide chat-based interfaces to a handful of services such as weather forecasts, movies, and restaurants. Similar to traditional CIs, these systems provide access to only a limited set of services.

Several research efforts have focused on developing parallel webs that facilitate voice-based access via conversational interfaces. The Spoken Web [11] creates a parallel telephone-based "web" of information based on audio recordings. VoiceXML [3] is a W3C standard for specifying interactive voice dialogues, analagous to HTML for GUI applications. The Semantic Web [3] will enable agents to semantically interact with information on the web. However, for the time being, the vast majority of services today are only accessible via the web. Therefore, we believe that conversational interfaces can and should be built on top of existing websites to give them the broadest impact.

# 3. WEB-BASED CONVERSATIONAL INTERFACES

The architecture of a traditional CI includes three main components: understanding the user's *input*, managing a *dialogue* with the user, and generating *output*. In traditional CIs, all of these components leverage the structure and contents of a database. For example, the input component can use the names of tables and columns in the database schema in order to understand a user's query for "French restaurants near downtown" as referring to cuisine=French, object=restaurants, and location=downtown.

When building CIs on top of web applications, one cannot take advantage of that type of information. However, we argue that the web provides different types of information that make it possible to build CIs effectively. Recent advances in algorithms for understanding and manipulating web applications make this possible.

## 3.1 Input

A key difference between CIs for databases and CIs for the web is that the task of searching a database is fairly well structured. Therefore the words in users' utterances can be interpreted relative to the information available in the database and database schema. On the web, however, the set of tasks is much broader. User input is not constrained to be on a specific topic. How can a CI make sense of what the user is saying and map it to a particular information need or requested task?

Ultimately the problem of understanding a user's input reduces to the problem of understanding what they are asking the computer to do. The challenge of a CI system for the web is to interpret that input and map it to something that can be done on the web. Typically, this process involves selecting a website and deciding what to do on that website.

Operating a user interface is similar to a conversation: the system presents some information, the user selects an option from among the choices presented, and the system responds with both the reaction to the user's command, along with new options for what to do next. On the web, this conversation is conducted through web pages and interactions on web pages: the system generates some HTML content and the user responds by clicking buttons and links and typing into textboxes. The system responds by generating a new web page, and so on.

Accessible technologies for visually impaired users illustrate one extreme of mapping from web-based interfaces into a conversational interface [5]. Screenreaders orally read out the content on the web page, top to bottom, and the user can indicate which link to follow or can navigate through the DOM structure of the page using keyboard commands. However, since the interaction is at the very low level of individual elements on the page, the conversation is very lengthy and laborious.

We believe it ought to be possible to create conversational interfaces for web applications that function at a higher level than the individual elements on the page.

One insight is that the input widgets in a web application (buttons, links, etc.) usually correspond to goals the user can express in conversation. For example, a user who wants to check in for a flight on an airline website will likely click the "Check In" button. The TrailBlazer system [4] made use of this insight in a system which, given a goal expressed in natural language, guided blind users through a sequence of steps to accomplish that goal.

A different approach, taken by CoCo [12], is to allow end user programmers to define a library of tasks that can be accomplished via web applications. The user's query is interpreted relative to the available tasks and the appropriate task model is selected for the next phase of the conversation.

Yet we can also imagine more ambitious ways to map web applications to a conversation. Imagine the situation where you call a friend for help while travelling, and ask your friend to do something for you. If you know what you need to do, such as checking the train schedules for a particular route, your friend can interpret the contents of that website and perform the task for you, as you provide the information necsesary to complete that task.

What if we could create an agent that could take the place of that friend, interpreting the contents of the website and relaying the relevant information back to you so that you can guide the agent through the task? The friend/agent does not need to be an expert in performing the task; they only need to be able to communicate the contents of websites efficiently to a human and decide what to do next. Therefore, through conversation, it may be possible to enable people to direct agents to perform arbitrary tasks on web applications.

## 3.2 Dialogue

Once we understand what the user wants to do, the next phase of conversation is to guide them through performing their task. In traditional CIs, this phase includes refining search results, presenting intermediate options, and helping the user navigate to the information they are seeking.

More generally, for web-based applications where the set of tasks is more varied than the classic database search problem, we need to have a task model in order to guide the user through performing their task, the same way traditional CIs guide users through finding information. Yet how are we going to get task models for arbitrary web applications?

If those task models already existed, building CIs would be easier. Research on model-based UI design [17] argues that task models should be the basis for designing and developing applications. If all designers and developers followed this

[1] http://siri.com

[2] http://guru.googlelabs.com

[3] http://en.wikipedia.org/wiki/VoiceXML

approach, we would have the task models we needed.

Yet even without explicit task models, the design of a web application imparts a lot of information which can be leveraged to infer task models. Because web applications are designed for humans to understand and use, their user interfaces codify designers' beliefs about what can be done using the application – their task models. By examining the user interfaces, it should be possible to reverse engineer the task models of applications.

The structure of web applications often gives a clue to their semantics. Well-designed web applications use consistent visual cues to enable users to quickly understand the site's contents [20]. For example, link text provides a description of the link destination, such as "flights" or "hotels". Moreover, links with similar functionality tend to be grouped together, such that links for "books" and "movies" and "CDs" are displayed physically close to each other. These design practices encode valuable information about the categories of objects that can be manipulated using this application. For example, a conversational interface should be able to analyze the navigation bar on the Google front page and determine that this website is organized according to Web, Images, Videos, Maps, News, and more.

Similarly, buttons on web pages often indicate actions that can be done using the application, such as "find a flight" or "search prices" or "download music". These buttons provide verbal cues for the types of tasks possible on the site.

More generally, websites today make use of a set of common UI patterns, such as user registration/login, search results, and shopping carts. For example, the search result pattern typically involves the user entering a query into a search textbox, to which the system responds with a list of summarized search results. The user clicks on one of the search results, and the system shows a page with more details about that specific item. Similarly, a shopping cart pattern has affordances to add/remove/update items in the cart and invoke the checkout process. While the actual rendering of these patterns changes from site to site, they share a common interaction flow that can be recognized.

The FindThis system [16] leveraged the search result pattern to provide a limited conversational interface for retrieving item-property information from e-commerce sites. With more research, we believe it should be possible to infer general task models from arbitrary web sites in order to support dialogue management in CIs.

### *3.2.1 Usage information*

We believe that what people have done in the past is a good predictor for what they will do in the future. Therefore, by examining past web interaction history, we ought to be able to build good task models.

The prevalence of web interaction logging makes it very easy to capture user interaction with web applications. Systems such as Coremetrics[4] and CoScripter Reusable History [13] capture fine-grained user activity at the level of interaction with elements on each web page. This user interaction data could be analyzed to identify common paths through the application. If many users tend to follow the same navigational path through the application, this "path well travelled" could form the basis of a task model.

For example, the CoCo system [12] mined personal usage logs to suggest automated web tasks that could be invoked through a conversational interface.

In the future, we envision an agent that has full knowledge of what the user has done in the past, and which can access the web on the user's behalf. The user should be able to converse with this agent in order to ask it to do tasks on the web, using that user's interaction logs as the basis for performing those tasks. When the user makes a request of the agent, the agent should be able to search through the user's past interaction history and identify interaction sequences that could be reused in order to accomplish the same task again. Through conversation, the agent can determine which steps to take to accomplish the desired goal, perhaps customize them to the user's current need.

### *3.2.2 Parameters*

Part of the dialogue management process in traditional CIs is to help users refine their query by providing additional search parameters or filters. The dialogue management component in traditional CIs includes strategies for prompting the user to provide this information based on current results.

However, in the web domain, tasks are not as structured as the traditional database search task. Instead of trying to find information, users may be applying for a scholarship, registering for a conference, or booking a vacation.

On the web, completing these tasks involves form filling. The web application designer has decided which parameters are needed for completion of a task, and has created a form to collect those inputs from the user.

These forms are natural points to engage in dialogue with the user to solicit their input. For example, the explicit task models authored for the CoCo system [12] included parameters which caused the dialogue manager to prompt the user for the values of those parameters before invoking the task.

More generally, we imagine a system that can automatically map a web form to a conversational dialogue with the user. For each input field in the form, the system can turn that into a dialogue prompt depending on the type of the input field (e.g., "what date would you like to leave?" and "what airport are you departing from?" and "do you want one-way or round trip?").

## 3.3 Output

In the traditional database search paradigm for conversational interfaces, output is typically represented as a collection of results, and the CI is responsible for communicating those results to the user. Since the results generally conform to a known schema, that schema can be used to guide the presentation of the results, such as by responding differently depending on the distribution of data values [18].

However, to build a CI based on a web application, the CI must be able to interpret results embedded in a web page and communicate them back to the user.

One approach to this problem is to create tools that let programmers manually specify how to extract content from web pages for use in the CI, along the lines of Highlight [15] (while designed for the creation of mobile interfaces, similar techniques could apply to CIs). Similarly, CoCo [12] let end user programmers author explicit "clip" commands to extract specific regions of content from web pages.

Another approach is to build on algorithms that automatically extract semantic structure from web pages, in the same way that Sifter [10] or Vegemite [14] automatically extract search results and repetitive content. On many sites, the

[4] http://coremetrics.com

content is output from a database using templates, leading to algorithms that can automatically infer the template and extract the database values [2]. Once extracted, search results could be communicated to the user via the CI using existing techniques for presenting structured result sets.

Alternatively, CIs could leverage work on screen scrapers such as d.mix [9], Kapow[5], or Yahoo! Pipes[6]. It is natural to expect that as this research area matures, we will start seeing technology that can extract structured data from arbitrary web pages for use in CIs.

## 4. CONCLUSION

We have proposed a new way to build conversational interfaces: by building them on top of existing web applications. Advances in task modeling, web usage mining, information extraction, and end user programming raise the possibility of creating general-purpose conversational interfaces for any task that can be accomplished on the web.

Unlike traditional CI architectures, the components of web-based CIs will be highly interdependent. Input, output, and dialogue models are tightly coupled. Task models will be used to interpret input utterances in addition to driving the dialogue. Components for extracting output from web pages and mapping user input to web interactions will be used for intermediate steps in the dialogue, such as selecting from a list of search results. Through a combination of existing technology, and new research, we believe web-based CIs are within reach.

## 5. REFERENCES

[1] J. F. Allen, L. K. Schubert, G. Ferguson, P. Heeman, C. H. Hwang, T. Kato, M. Light, N. G. Martin, B. W. Miller, M. Poesio, and D. R. Traum. The TRAINS Project: A case study in building a conversational planning agent. *Journal of Experimental and Theoretical AI*, 7:7–48, 1994.

[2] A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *Proc. 2003 ACM SIGMOD Intl Conf on Management of Data*, SIGMOD '03, pages 337–348, New York, NY, USA, 2003. ACM.

[3] T. Berners-lee and J. Hendler. The semantic web. *Scientific American*, 284:34–43, 2001.

[4] J. P. Bigham, T. Lau, and J. Nichols. TrailBlazer: Enabling blind users to blaze trails through the web. In *Proc. 14th Intl Conf on Intelligent User Interfaces*, IUI '09, pages 177–186, New York, NY, USA, 2009.

[5] J. P. Bigham, C. M. Prince, and R. E. Ladner. WebAnywhere: A screen reader on-the-go. In *Proc. 2008 Intl Cross-disciplinary Conference on Web Accessibility*, W4A '08, pages 73–82, New York, NY, USA, 2008.

[6] J. R. Glass, E. Weinstein, D. S. Cyphers, J. Polifroni, G. Chung, and M. Nakano. A framework for developing conversational user interfaces. In *CADUI*, pages 347–358, 2004.

[7] A. L. Gorin, G. Riccardi, and J. H. Wright. How may I help you? *Speech Commun.*, 23:113–127, Oct 1997.

[8] B. J. Grosz. Transportable natural-language interfaces: problems and techniques. In *Proc. 20th Annual Meeting on Association for Computational Linguistics*, ACL '82, pages 46–50, Stroudsburg, PA, USA, 1982.

[9] B. Hartmann, L. Wu, K. Collins, and S. R. Klemmer. Programming by a sample: rapidly creating web applications with d.mix. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*, UIST '07, pages 241–250, New York, NY, USA, 2007. ACM.

[10] D. F. Huynh, R. C. Miller, and D. R. Karger. Enabling web browsers to augment web sites' filtering and sorting functionalities. In *Proc 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, pages 125–134, New York, NY, USA, 2006. ACM.

[11] A. Kumar, N. Rajput, D. Chakraborty, S. K. Agarwal, and A. A. Nanavati. WWTW: the world wide telecom web. In *Proc. 2007 Workshop on Networked Systems for Developing Regions*, NSDR '07, pages 7:1–7:6, New York, NY, USA, 2007. ACM.

[12] T. Lau, J. Cerruti, G. Manzato, M. Bengualid, J. P. Bigham, and J. Nichols. A conversational interface to web automation. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, pages 229–238, New York, NY, USA, 2010. ACM.

[13] I. Li, J. Nichols, T. Lau, C. Drews, and A. Cypher. Here's What I Did: Sharing and Reusing Web Activity with ActionShot. In *Proc. 28th Intl Conf on Human Factors in Computing Systems*, CHI '10, pages 723–732, New York, NY, USA, 2010. ACM.

[14] J. Lin, J. Wong, J. Nichols, A. Cypher, and T. A. Lau. End-user Programming of Mashups with Vegemite. In *Proc 14th Intl Conf on Intelligent User Interfaces*, IUI '09, pages 97–106, New York, NY, USA, 2009. ACM.

[15] J. Nichols, Z. Hua, and J. Barton. Highlight: a system for creating and deploying mobile web applications. In *Proc 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 249–258, New York, NY, USA, 2008. ACM.

[16] I. Okoye, J. Mahmud, T. Lau, and J. Cerruti. Find This for Me: Mobile Information Retrieval on the Open Web. In *Proc 16th Intl Conf on Intelligent User Interfaces*, IUI '11, pages 3–12, New York, NY, USA, 2011. ACM.

[17] F. Paterno. *Model-Based Design and Evaluation of Interactive Applications.* 2000.

[18] J. Polifroni, G. Chung, and S. Seneff. Towards automatic generation of mixed-initiative dialog systems from web content. In *Proc. of Eurospeech*, 2003.

[19] A.-M. Popescu, O. Etzioni, and H. Kautz. Towards a theory of natural language interfaces to databases. In *Proc. 8th Intl Conf on Intelligent User Interfaces*, IUI '03, pages 149–157, New York, NY, USA, 2003.

[20] L. Rosenfeld and P. Morville. *Information Architecture for the World Wide Web.* O'Reilly, 2002.

[21] V. Zue and J. Glass. Conversational interfaces: advances and challenges. *Proceedings of the IEEE*, 88(8):1166 –1180, Aug 2000.

[5] http://kapowsoftware.com

[6] http://pipes.yahoo.com